comprising a computer usable medium having computer readable program code thereon, including program code which:

stores an object of a particular type in one of a plurality of logical partitions in the heap dependent on a predefined category for the type;

searches one of the logical partitions for referenced objects; and

reclaims non-referenced objects stored in the searched logical partition.

a5

Amendments to the claims are indicated in the attached "Marked Up Version of Amendments" (pages i – iv).

REMARKS

Claims 1-26 are pending in the application. Claim 16 has been objected to because of informalities. Claims 1, 9, 17, 25 and 26 are rejected under 35 U.S.C. § 102(e) as being deemed anticipated by U.S. Publication No. 2002/0166116 (Eidt). Claims 2-3, 6-8, 10-11, 14-16, 18-19 and 22-24 are rejected under 35 U.S.C. § 103(a) as being deemed unpatentable over U.S. Publication No. 2002/0166116 (Eidt) in view of U.S. Patent No. 5,485,613 (Engelstad et al.). Claims 4-5, 12-13 and 20-21 are rejected under 35 U.S.C. § 103(a) as being deemed unpatentable over U.S. Publication No. 2002/0166116 (Eidt) in view of U.S. Patent No. 5,485,613 (Engelstad et al.) as applied to Claims 1-3, 6-11, 14-19 and 22-26 above, and further in view of U.S. Patent No. 6,047,295 (Endicott et al.). Of the Claims 1, 9, 25 and 26 are independent. The application, as amended and argued herein, is believed to overcome the rejections.

With regard to the objection to Claim 16, the Applicant has amended Claim 16 as requested by the Examiner. Reconsideration of the objection to Claim 16 in favor of allowance is respectfully requested.

None of the cited references teaches or suggests at least the Applicant's claimed "object allocation routine which stores an object of a particular type in one of a plurality of logical partitions in the heap dependent on a predefined category for the object type." (*See* Claims 1, 9, 17, 25 and 26.) The cited references merely discuss methods for searching the heap for collectable objects stored in the heap. (*See* Eidt [0072]; Endicott, Fig. 3, 74; Englestad, Abstract.) In contrast, the Applicant's claimed "object allocation routine" stores an object in one of a plurality of logical partitions in the heap "dependent on a predefined category for the object type". (See Fig 4, 408 in the Applicant's application as originally filed.) By storing objects based on a predefined category (hot or cold) for the type of object, the efficiency of collecting non-referenced objects is increased because the collection routine only searches "one of the logical partitions" in the heap storing objects of a particular type and the collector "reclaims non-referenced objects stored in the searched logical partition".

Claims 2-8 are dependent on Claim 1, Claims 10-16 are dependent on Claim 9, Claims 18-24 are dependent on Claim 17, and thus include this limitation over the cited art.

Therefore, separately or in combination, Eidt, Engelstad and Endicott fail to teach or suggest the Applicant's claimed invention. Reconsideration of the rejections under 35 U.S.C. § 102(e) and 35 U.S.C. § 103(a) is respectfully requested.

CONCLUSION

In view of the above amendments and remarks, it is believed that all claims (Claims 1-26) are in conditions for allowance, and it is respectfully requested that the application be passed to issue. If the Examiner feels that a telephone conference would expedite prosecution of this case, the Examiner is invited to call the undersigned at (978) 341-0036.

Respectfully submitted,

HAMILTON, BROOK, SMITH & REYNOLDS, P.C.

By [illegible]
Caroline M. Fleming, Esq.
Registration No.: 45,566
Telephone: (978) 341-0036
Facsimile: (978) 341-0136

Concord, MA 01742-9133
Dated: 6/30/03

OIPE JC42
JUL 0 3 2003
PATENT & TRADEMARK OFFICE

MARKED UP VERSION OF AMENDMENTS

Claim Amendments Under 37 C.F.R. § 1.121(c)(1)(ii)

1. (Amended) A collector for collecting non-referenced objects stored in a heap by a program executing in a computer system comprising:

   an object allocation routine which stores an object of a particular type in one of a plurality of [spaces] logical partitions in the heap dependent on a predefined category for the object type; and

   a collection routine which searches one of the [spaces] logical partitions for referenced objects and reclaims non-referenced objects stored in the searched [space] logical partition.

3. (Amended) The collector as claimed in Claim 2 wherein upon determining that a hot [space] logical partition is full, the collection routine searches a cold logical partition [space] and the hot [space] logical partition for referenced objects and moves referenced objects of the hot category stored in the hot [space] logical partition to the cold [space] logical partition.

4. (Amended) The collector as claimed in Claim 2 wherein the sample and partition further comprises:

   a write barrier elimination routine, which eliminates a write barrier for an intergenerational pointer between an object stored in a hot [space] logical partition and an object stored in a cold logical partition [space].

8. (Amended) The collector as claimed in Claim 2 wherein the sample and partition routine partitions the heap to minimize intergenerational pointers between a hot [space] logical partition and a cold logical partition [space].

9. (Amended) A collector for collecting non-referenced objects stored in a heap by a program executing in a computer system comprising:

means for storing an object of a particular type in one of a plurality of logical partitions[spaces] in the heap dependent on a predefined category for the object type;

means for searching one of the logical partitions [spaces] for referenced objects; and

means for reclaiming non-referenced objects stored in the searched logical partition [space].

10. (Amended) The collector as claimed in Claim 9 further comprising:

means for partitioning the heap into a cold logical partition [space] and a hot [space] logical partition by defining a category of an object stored in the heap to be hot or cold.

11. (Amended) The collector as claimed in Claim 10 wherein upon determining that a hot [space] logical partition is full, the means for searching searches a cold logical partition [space] and the hot logical partition [space] for referenced objects and moves referenced objects stored in the hot [space] logical partition to [a] the cold [space] logical partition.

12. (Amended) The collector as claimed in Claim 10 wherein the means for partitioning further comprises:

means for eliminating a write barrier for an intergenerational pointer between an object stored in the hot [space] logical partition and an object stored in the cold [space] logical partition.

16. (Amended) The collector as claimed in Claim [9] 10 wherein the means for partitioning partitions the heap to minimize intergenerational pointers between the hot [space] logical partition and the cold [space] logical partition.

17. (Amended) A method for collecting non-referenced objects stored in a heap by a program executing in a computer system comprising the steps of:

storing an object of a particular type in one of a plurality of [spaces] logical partitions in the heap dependent on a predefined category for the object type;

searching one of the [spaces] logical partitions for referenced objects; and

reclaiming non-referenced objects stored in the searched [space] logical partition.

18. (Amended) The method as claimed in Claim 17 further comprising the step of:

partitioning the heap into a cold [space] logical partition and a hot [space] logical partition by defining hot [space] logical partition objects and cold [space] logical partition objects.

19. (Amended) The method as claimed in Claim 18 wherein upon determining that the hot [space] logical partition is full, the step of reclaiming further comprises the step of:

moving referenced objects stored in the hot [space] logical partition to [a] the cold [space] logical partition.

20. (Amended) The method as claimed in Claim 18 wherein the step of partitioning further comprises the step of:

eliminating a write barrier for an intergenerational pointer between an object stored in the hot [space] logical partition and an object stored in the cold [space] logical partition.

24. (Amended) The method as claimed in Claim 18 wherein the step of partitioning partitions the heap to minimize intergenerational pointers between the hot [space] logical partition and the cold [space] logical partition.

25. (Amended) A computer system comprising:

a central processing unit connected to a memory bus by a system bus;

an I/O system, connected to the system bus by a bus interface; and

a collector for collecting non-referenced objects stored in a heap by a program executing in a computer system, the collector:

storing an object of a particular type in one of a plurality of [spaces] logical partitions in the heap dependent on a predefined category for the type;

searching one of the [spaces] logical partitions for referenced objects; and

reclaiming non-referenced objects stored in the searched [space] logical partition.

26. (Amended) A computer program product for collecting non-referenced objects stored in a heap by a program executing in a computer system, the computer program product comprising a computer usable medium having computer readable program code thereon, including program code which:

stores an object of a particular type in one of a plurality of [spaces] logical partitions in the heap dependent on a predefined category for the type;

searches one of the [spaces] logical partitions for referenced objects; and

reclaims non-referenced objects stored in the searched [space] logical partition.